MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 3, 2021/2022

ECP2036 – MICROPROCESSOR SYSTEM AND INTERFACING

(ME)

09 AUGUST 2022
9:00AM - 11:AM
(2 Hours)

INSTRUCTIONS TO STUDENT

1. This Question paper consists of 9 pages including cover page and appendix with 4 Questions only.

2. Attempt **ALL** questions. All questions carry equal marks and the distribution of the marks for each question is given.

3. Please write all your answers in the Answer Booklet provided.

Question 1

(a) With the aid of a suitable diagram, describe the major components of a computer. Briefly discuss the function of each component.

[6 marks]

(b) Highlight the important features that distinguish microcontroller from microprocessor based on the hardware architecture and applications perspective.

[6 marks]

(c) The capacity of a memory chip is specified as 64K*8 bits. Find out the data bus and the address bus for the chip. A memory block has 10 address lines and 16 data lines, find its capacity.

[5+3 marks]

(d) Name any five products each that likely to use a microcontroller and/or a microprocessor.

[5 marks]

Continued...

Question 2

(a) An 8051-microcontroller based system is to be designed with the requirement of external 32Kbytes ROM and 32Kbytes RAM, with start-up codes stored in ROM. Given an available 8Kbytes program memory and 16Kbytes data memory blocks, answer the following questions:

i. What is the size of the data bus of this system?
[2 marks]

ii. Identify the number of ROM and RAM blocks needed for the system.
[3 marks]

iii. Calculate the address lines needed by each of these ROM and RAM blocks.
[3 marks]

iv. With ROM occupying the first 32Kbytes of memory space, determine the starting and ending addresses for each memory blocks.
[3 marks]

(b) Fill up the Table 1 below for the value of the accumulator (A) and the carry flag (C) after the execution of each instruction.
[6 marks]

Table 1

	A	C
MOV R1,#0AAH	00H	0
MOV A, #55H		
ADD A, R1		
ADD A, #1		

(c) A microprocessor is fetching an instruction (2BH) from the memory location (10ABH). Draw the timing diagram of the address & data buses and any other control signal involved.
[8 marks]

Continued...

Question 3

(a) Using the MCS-51 opcode map as given in the Appendix A.

```
        ORG 0100H
DELAY:  MOV R0, A
LOOP1:  MOV R1, #250
LOOP2:  MOV R2, #250
LOOP3:  NOP
        NOP
        DJNZ R2, LOOP3
        DJNZ R1, LOOP2
        DJNZ R0, LOOP1
```

i. Convert the above DELAY subroutine into the corresponding machine codes.

[5 marks]

ii. Determine the total number of bytes of the above DELAY subroutine.

[1 marks]

iii. Determine the total execution time of the above DELAY subroutine.

[1 marks]

iv. List down the addressing modes of the instructions used in the above DELAY subroutine.

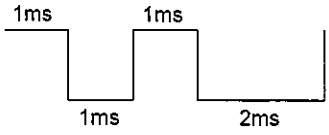
[2 marks]

v. Write a simple assembly program to call the above DELAY subroutine to delay for 30 minutes.

[2 marks]

(b) Write an 8051 assembly program to generate the following signal continuously from port P1.0 using Timer 1. Find also the adjustments for timer-reload values to generate the signal as accurately as possible.

[7 marks]



(c) Write an 8051 assembly program to send the input value from Port 0, 1 or 2 based on the byte value it receives from the serial port. For example, if the serial port receives 0, the program should send out the input value of Port 0; if serial port receives 1, the program should send out the input value of Port 1. Assume the serial port is configured as 8-bit UART at 9600 baud rate.

[7 marks]

Continued...

Question 4

(a) In the Figure Q4a, assume that the $\overline{\text{INT0}}$ pin is connected to a switch that is normally high. Whenever it goes low, it should turn on an LED connected to P1.0 for 1 second using Timer 1. The LED is normally off, the LED should stay on as long as the switch is pressed low. Write a program to perform this task using External 0 and Timer 1 interrupts.
[10 marks]

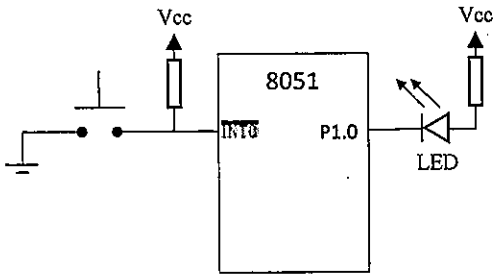


Figure Q4a

(b) A numerical keypad and a common cathode 7-segment display with the following arrangement are to be interfaced to an 8051 microcontroller (Figure Q4b).

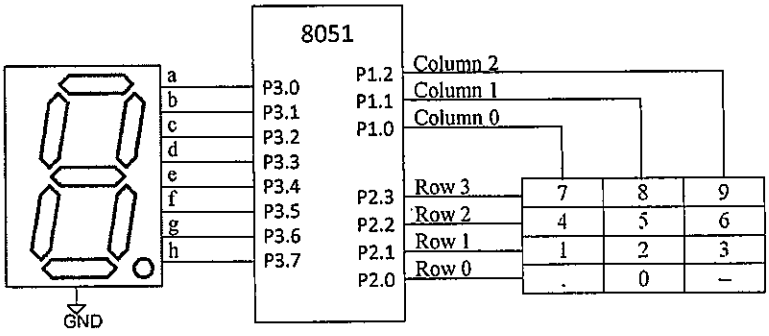


Figure Q4b

i. Write a subroutine to continuously scan the keypad and return the position of the column and row of the key pressed into R1 and R2 respectively.
[8 marks]

ii. Given the row information is stored in R1, while the column information is stored in R2. Write a subroutine to display the key pressed character ('.', '–', '0', '1'…'9') to the 7-segment display.
[7 marks]

End of Paper

Appendix A: Opcode Map

byte Instruction operands cycle	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1B **NOP** 1C	3B **JBC** bit, rel 2C	3B **JB** bit, rel 2C	3E **JNB** bit, rel 2C	2B **JC** rel 2C	2B **JNC** rel 2C	2B **JZ** rel 2C	2B **JNZ** rel 2C	2E **SJMP** rel 2C	3B **MOV** DPTR, #data16 2C	2B **ORL** C, /bit 2C	2B **ANL** C, /bit 2C	2B **PUSH** dir 2C	2B **POP** dir 2C	1B **MOVX** A, @DPTR 2C	1B **MOVX** @DPTR, A
1	2B **AJMP** (P0) 2C	2B **ACALL** (P0) 2C	2B **AJMP** (P1) 2C	2B **ACALL** (P1) 2C	2B **AJMP** (P2) 2C	2B **ACALL** (P2) 2C	2B **AJMP** (P3) 2C	2B **ACALL** (P3) 2C	2B **AJMP** (P4) 2C	2B **ACALL** (P4) 2C	2B **AJMP** (P5) 2C	2B **ACALL** (P5) 2C	2B **AJMP** (P6) 2C	2B **ACALL** (P6) 2C	2B **AJMP** (P7) 2C	2B **ACALL** (P7) 2C
2	3B **LJMP** addr16 2C	3B **LCALL** addr16 2C	1B **RET** 2C	1E **RETI** 2C	2B **ORL** dir, A 1C	2B **ANL** dir, A 1C	2B **XRL** dir, A 1C	2B **ORL** C, bit 2C	2E **ANL** C, bit 2C	2B **MOV** bit, C 2C	2B **MOV** C, bit 1C	2B **CPL** bit 1C	2B **CLR** bit 1C	2B **SETB** bit 1C	1B **MOVX** A, @R0 2C	1B **MOVX** @R0, A
3	1B **RR** A 1C	1B **RRC** A 1C	1B **RL** A 1C	1E **RLC** A 1C	3B **ORL** dir, #data 2C	3B **ANL** dir, #data 2C	3B **XRL** dir, #data 2C	1B **JMP** @A+DPTR 2C	1B **MOVC** A, @A+PC 2C	1B **MOVC** A, @A+DPTR 2C	1B **INC** DPTR 2C	1B **CPL** C 1C	1B **CLR** C 1C	1B **SETB** C 1C	1B **MOVX** A, @R1 2C	1B **MOVX** @R1, A
4	1B **INC** A 1C	1B **DEC** A 1C	2B **ADD** A, #data 1C	2B **ADDC** A, #data 1C	2B **ORL** A, #data 1C	2B **ANL** A, #data 1C	2B **XRL** A, #data 1C	2B **MOV** A, #data 1C	1E **DIV** AB 4C	2B **SUBB** A, #data 1C	1B **MUL** AB 4C	3B **CJNE** A, #data, rel 2C	1B **SWAP** A 1C	1B **DA** A 1C	1B **CLR** A 1C	1B **CPL** A 1C
5	2B **INC** dir 1C	2B **DEC** dir 1C	2B **ADD** A, dir 1C	2B **ADDC** A, dir 1C	2B **ORL** A, dir 1C	2B **ANL** A, dir 1C	2B **XRL** A, dir 1C	3B **MOV** dir, #data 2C	3B **MOV** dir, dir 2C	2B **SUBB** A, dir 1C		3B **CJNE** A, dir, rel 2C	2B **XCH** A, dir 1C	3B **DJNZ** dir, rel 2C	2B **MOV** A, dir 1C	2B **MOV** dir, A 1C
6	1B **INC** @R0 1C	1B **DEC** @R0 1C	1B **ADD** A, @R0 1C	1B **ADDC** A, @R0 1C	1B **ORL** A, @R0 1C	1B **ANL** A, @R0 1C	1B **XRL** A, @R0 1C	2B **MOV** @R0, #data 1C	2E **MOV** dir, @R0 2C	1B **SUBB** A, @R0 1C	2B **MOV** @R0, dir 2C	3B **CJNE** @R0, #data,rel 2C	1B **XCH** A, @R0 1C	1B **XCHD** A, @R0 1C	1B **MOV** A, @R0 1C	1B **MOV** @R0, A 1C
7	1B **INC** @R1 1C	1B **DEC** @R1 1C	1B **ADD** A, @R1 1C	1B **ADDC** A, @R1 1C	1B **ORL** A, @R1 1C	1B **ANL** A, @R1 1C	1B **XRL** A, @R1 1C	2B **MOV** @R1, #data 1C	2E **MOV** dir, @R1 2C	1B **SUBB** A, @R1 1C	2B **MOV** @R1, dir 2C	3B **CJNE** @R1, #data,rel 2C	1B **XCH** A, @R1 1C	1B **XCHD** A, @R1 1C	1B **MOV** A, @R1 1C	1B **MOV** @R1, A 1C
8	1B **INC** R0 1C	1B **DEC** R0 1C	1B **ADD** A, R0 1C	1B **ADDC** A, R0 1C	1B **ORL** A, R0 1C	1B **ANL** A, R0 1C	1B **XRL** A, R0 1C	2B **MOV** R0, #data 1C	2B **MOV** dir, R0 2C	1B **SUBB** A, R0 1C	2B **MOV** R0, dir 2C	3B **CJNE** R0, #data, rel 2C	1B **XCH** A, R0 1C	2B **DJNZ** R0, rel 2C	1B **MOV** A, R0 1C	1B **MOV** R0, A 1C
9	1B **INC** R1 1C	1B **DEC** R1 1C	1B **ADD** A, R1 1C	1B **ADDC** A, R1 1C	1B **ORL** A, R1 1C	1B **ANL** A, R1 1C	1B **XRL** A, R1 1C	2B **MOV** R1, #data 1C	2B **MOV** dir, R1 2C	1B **SUBB** A, R1 1C	2B **MOV** R1, dir 2C	3B **CJNE** R1, #data, rel 2C	1B **XCH** A, R1 1C	2B **DJNZ** R1, rel 2C	1B **MOV** A, R1 1C	1B **MOV** R1, A 1C
A	1B **INC** R2 1C	1B **DEC** R2 1C	1B **ADD** A, R2 1C	1B **ADDC** A, R2 1C	1B **ORL** A, R2 1C	1B **ANL** A, R2 1C	1B **XRL** A, R2 1C	2B **MOV** R2, #data 1C	2B **MOV** dir, R2 2C	1B **SUBB** A, R2 1C	2B **MOV** R2, dir 2C	3B **CJNE** R2, #data, rel 2C	1B **XCH** A, R2 1C	2B **DJNZ** R2, rel 2C	1B **MOV** A, R2 1C	1B **MOV** R2, A 1C
B	1B **INC** R3 1C	1B **DEC** R3 1C	1B **ADD** A, R3 1C	1B **ADDC** A, R3 1C	1B **ORL** A, R3 1C	1B **ANL** A, R3 1C	1B **XRL** A, R3 1C	2B **MOV** R3, #data 1C	2B **MOV** dir, R3 2C	1B **SUBB** A, R3 1C	2B **MOV** R3, dir 2C	3B **CJNE** R3, #data, rel 2C	1B **XCH** A, R3 1C	2B **DJNZ** R3, rel 2C	1B **MOV** A, R3 1C	1B **MOV** R3, A 1C
C	1B **INC** R4 1C	1B **DEC** R4 1C	1B **ADD** A, R4 1C	1B **ADDC** A, R4 1C	1B **ORL** A, R4 1C	1B **ANL** A, R4 1C	1B **XRL** A, R4 1C	2B **MOV** R4, #data 1C	2B **MOV** dir, R4 2C	1B **SUBB** A, R4 1C	2B **MOV** R4, dir 2C	3B **CJNE** R4, #data, rel 2C	1B **XCH** A, R4 1C	2B **DJNZ** R4, rel 2C	1B **MOV** A, R4 1C	1B **MOV** R4, A 1C
D	1B **INC** R5 1C	1B **DEC** R5 1C	1B **ADD** A, R5 1C	1B **ADDC** A, R5 1C	1B **ORL** A, R5 1C	1B **ANL** A, R5 1C	1B **XRL** A, R5 1C	2B **MOV** R5, #data 1C	2B **MOV** dir, R5 2C	1B **SUBB** A, R5 1C	2B **MOV** R5, dir 2C	3B **CJNE** R5, #data, rel 2C	1B **XCH** A, R5 1C	2B **DJNZ** R5, rel 2C	1B **MOV** A, R5 1C	1B **MOV** R5, A 1C
E	1B **INC** R6 1C	1B **DEC** R6 1C	1B **ADD** A, R6 1C	1B **ADDC** A, R6 1C	1B **ORL** A, R6 1C	1B **ANL** A, R6 1C	1B **XRL** A, R6 1C	2B **MOV** R6, #data 1C	2B **MOV** dir, R6 2C	1B **SUBB** A, R6 1C	2B **MOV** R6, dir 2C	3B **CJNE** R6, #data, rel 2C	1B **XCH** A, R6 1C	2B **DJNZ** R6, rel 2C	1B **MOV** A, R6 1C	1B **MOV** R6, A 1C
F	1B **INC** R7 1C	1B **DEC** R7 1C	1B **ADD** A, R7 1C	1B **ADDC** A, R7 1C	1B **ORL** A, R7 1C	1B **ANL** A, R7 1C	1B **XRL** A, R7 1C	2B **MOV** R7, #data 1C	2B **MOV** dir, R7 2C	1B **SUBB** A, R7 1C	2B **MOV** R7, dir 2C	3B **CJNE** R7, #data, rel 2C	1B **XCH** A, R7 1C	2B **DJNZ** R7, rel 2C	1B **MOV** A, R7 1C	1B **MOV** R7, A 1C

Appendix B: Special Function Register (SFR)

Byte address	Bit address								
FF									
F0	F7	F6	F5	F4	F3	F2	F1	F0	B
E0	E7	E6	E5	E4	E3	E2	E1	E0	ACC
D0	D7	D6	D5	D4	D3	D2	D1	D0	PSW
B8	-	-	-	BC	BB	BA	B9	B8	IP
B0	B7	B6	B5	B4	B3	B2	B1	B0	P3
A8	AF	-	-	AC	AB	AA	A9	A8	IE
A0	A7	A6	A5	A4	A3	A2	A1	A0	P2
99	not bit addressable								SBUF
98	9F	9E	9D	9C	9B	9A	99	98	SCON
90	97	96	95	94	93	92	91	90	P1
8D	not bit addressable								TH1
8C	not bit addressable								TH0
8B	not bit addressable								TL1
8A	not bit addressable								TL0
89	not bit addressable								TMOD
88	8F	8E	8D	8C	8B	8A	89	88	TCON
87	not bit addressable								PCON
83	not bit addressable								DPH
82	not bit addressable								DPL
81	not bit addressable								SP
80	87	86	85	84	83	82	81	80	P0

Appendix C: Bit-Addressable Ram Locations

Byte Address	Bit Address							
2F	7F	7E	7D	7C	7B	7A	79	78
2E	77	76	75	74	73	72	71	70
2D	6F	6E	6D	6C	6B	6A	69	68
2C	67	66	65	64	63	62	61	60
2B	5F	5E	5D	5C	5B	5A	59	58
2A	57	56	55	54	53	52	51	50
29	4F	4E	4D	4C	4B	4A	49	48
28	47	46	45	44	43	42	41	40
27	3F	3E	3D	3C	3B	3A	39	38
26	37	36	35	34	33	32	31	30
25	2F	2E	2D	2C	2B	2A	29	28
24	27	26	25	24	23	22	21	20
23	1F	1E	1D	1C	1B	1A	19	18
22	17	16	15	14	13	12	11	10
21	0F	0E	0D	0C	0B	0A	09	08
20	07	06	05	04	03	02	01	00

Appendix D: TCON (Timer/Counter Control Register)

TF1	TR1	TF0	TR1	IE1	IT1	IE0	IT0
7	6	5	4	3	2	1	0

Appendix E: TMOD

GATE	C/T	M1	M0	GATE	C/T	M1	M0
7	6	5	4	3	2	1	0

Appendix F: IE (Interrupt Enable Register)

EA	--	ET2	ES	ET1	EX1	ET0	EX0
7	6	5	4	3	2	1	0

Appendix G: IP (Interrupt Priority Register)

--	--	PT2	PS	PT1	PX1	PT0	PX0
7	6	5	4	3	2	1	0

Appendix H: Interrupt Vectors

Interrupt	Flag	Vector Address
Reset	RST	0000H
External 0	IE0	0003H
Timer 0	TF0	000BH
External 1	IE1	0013H
Timer 1	TF1	001BH
Serial Port	RI or TI	0023H

Appendix I: SCON (Serial Port Control Register)

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
7	6	5	4	3	2	1	0